УДК 519.7+681.322

Активная защита программ от внедрения разрушающих модулей

Ревотюк М.П.*, Кишкевич А.П.
БГУИР*, ИП "Центр компетенции "Эвклид"

Предмет рассмотрения – объектно-ориентированная реализация системы активной защиты разрабатываемого программного обеспечения в вычислительных средах операционных систем семейства Windows 2000/XP. Функциональное замыкание интервалов существования защищаемых объектов, обеспечиваемое конструкторами и деструкторами классов языка C++, использовано для автоматической установки среды защиты, опирающейся на аппаратные автономные криптографических средства.

Исполняемые модули ответственных систем обработки информации, представленные загрузочными файлами или файлами динамически подключаемых библиотек с открытым форматом PE (Portable Executable) в среде операционных систем Windows уязвимы для ряда угроз несанкционированного использования – копирования, дизассемблирования, модификации и запуска. Один из приемов защиты – связь процесса исполнения программного кода с сервисами группы AAA(Авторизация, Аутентификация, Аудит), рекомендуемых для создания серверных приложений [1].

Однако технологии создания серверных приложений не защищают файлы исполняемых модулей от дизассемблирования, трассировки или других угроз, реализуемых после получения файла. Опора на стандартные системные средства администрирования также страдает уязвимостью – дискреционный метод разделения полномочий пользователей и политика безопасности по определению статичны. Кардинальным решением задач противодействия таким угрозам может быть криптографическая защита фрагментов кода.

Так как применение криптографии само по должно быть связано с состоянием аутентификации, то естественно образовать рекуррентную схему его связи с хотя бы одним

предшествующим и остальными доступными для фиксации состояниями (рис.1).

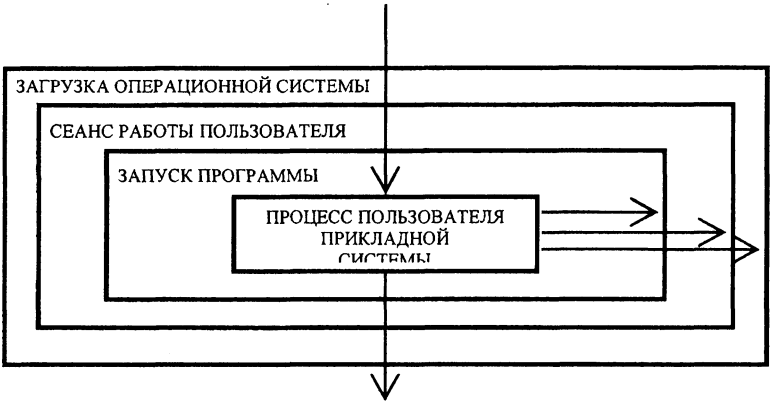

Рис. 1. Схема вложенности процессов активности пользователя

Архитектура операционных систем Windows NT/2000/XP/2003 включает достаточно полный набор интерфейсов для организации полного автоматического контроля вычислительной активности.

Используя криптографию с открытым ключом и реально доступные системные события, возможно до этапа инсталляции на ЭВМ построение динамической системы, привязанной к моменту аутентификации, функционирующей только при нулевых масках доступа к процессу лишь при предъявлении ключа зарегистрированного конечного пользователя (рис. 2) [1].

Файл программы выступает как контейнер для хранения скрытых блоков кода. Преобразование кода выполняется в последний момент непосредственно перед использованием в проекции на память. Критические по соображениям безопасности блоки прикладной программы должны быть указаны на уровне исходного кода [2].

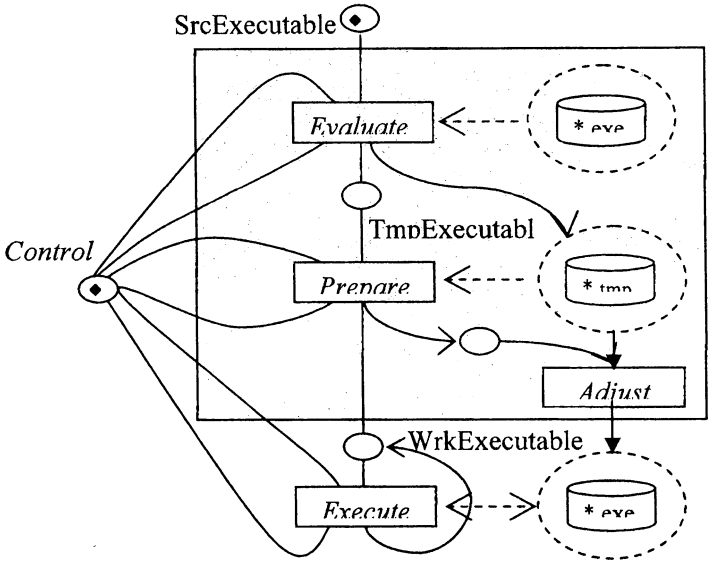

Рис. 2. Схема процесса установки слоев защиты

Выделенные блоки программы могут находиться в трех состояниях [3]:

SrcExecutableFile – исходное состояние в загрузочном модуле – результат работы редактора связей (компоновщика);

TmpExecutableFile – состояние актуализации привязки "файл – память", регистрируемое конструктором класса статической защиты при первом технологическом запуске программы на исполнение;

WrkExecutableFile – закрытое состояние, обнаруживаемое конструктором класса статической защиты в момент запуска защищенной версии программы.

Конструктор класса статической защиты обеспечивает перевод блоков защищаемого модуля в рабочее состояние непосредственно перед этапом исполнения функциональной части программы.

Легко заметить, что представленная схема соответствует расширенной сети Петри с очевидным поведением: после последовательной активизации переходов *Evaluate* (оценка обстановки) и *Prepare*(подготовка рабочей версии программы)

возможна лишь активизация перехода *Execute*(исполнение программы в защищенном режиме). После активизации перехода *Adjust*(подготовка к эксплуатации) уничтожаются следы процесса подготовки рабочей версии программы, включая код функций переходов *Evaluate*, *Prepare* и *Adjust*. Позиция *Control* соответствует необходимым условиям активизации перехода *Execute*, включая, например, значение ключа, считанного из аппаратного носителя.

Практическая реализация защиты не требует от разработчика функциональной части прикладной программы специальных действий. Декларация объекта защиты может выполняться независимым лицом, осведомленным о технологии обработки данных.

Таким образом, представляемая схема защиты на логическом уровне утилизирует потенциальные возможности автономных аппаратных средств криптографии для упреждающей проверки программного канала доступа к защищаемым данным.

Литература

1. Ревотюк М.П. Шаблоны систем обеспечения безопасности разрабатываемых программ в вычислительных средах с открытой архитектурой//Компьютерные технологии в обеспечении безопасности электронной информации: Материалы межд. конф.(Минск, 4-9 ноября 2002 г.) - Мн.: БелИСА, 2002. – с. 107-117.

2. Ховард М., Лебланк Д. Защищенный код/Пер. с англ. – М.: Издательско-торговый дом "Русская редакция", 2003.–704 с.

УДК 004.056.5

Генераторы подмножеств комбинаторных объектов

Кишкевич А.П., Ревотюк М.П.*
ИП "Центр компетенции "Эвклид", БГУИР*

Объект рассмотрения – вычислительные схемы решения комбинаторных задач методом перебора вариантов. В случае, когда множество вариантов порождается алгоритмически, возникает проблема построения их подмножеств с целью независимого анализа на вычислительной сети.